REMARKS

Claims 1-18 and 19-29 are pending in the present application. Reconsideration of the claims is respectfully requested.

I. 35 U.S.C. § 103, Obviousness

The Office Action rejects claims 1-18 and 20-29 under 35 U.S.C. § 103 as being unpatentable over *Densmore* (US Patent No. 6,591,305) in view of *Gosling et al.* (EP 0810524). This rejection is respectfully traversed.

With respect to claim 1, the Office Action states:

> As per claims 1, Densmore discloses a method for extending the capabilities of a web server, comprising the steps of:
>
> - sending a request from a client to the web server, the request identifying requested content and including addresses for a plurality of code module modules needed to service the request (col. 4, lines 42-53, and col. 6, lines 36-50);
> - receiving the requested content at the web server (col. 4, lines 42-53, and col. 6, lines 36-50); and
> - applying the installed plurality of code modules sequentially to the requested content (col. 8, lines 62-67, and col. 9, lines 1-4).
>
> However, Densmore does not explicitly disclose:
>
> - if the given code module is unavailable at the web server, having the web server use a corresponding address to request the given code module from a publishing server; and
> - installing the given code module at the web server.
>
> Gosling discloses a method and apparatus for operating a local server computer of a client-server network comprising:
>
> - if the given code module is unavailable at the web server, having the web server use a corresponding address to request the given code module from a publishing server (pg. 3, lines 25-30); and
> - installing the given code module at the web server (pg. 3, lines 25-34).
>
> Therefore, it would have been obvious to one of ordinary skill in the art to combine the teachings of Densmore and Gosling by requesting a code module from the publishing sever if it is unavailable at the web server allowing a user to receive the requested content in a timely and efficient manner.

Office Action, dated September 10, 2003. Applicant respectfully disagrees. *Densmore* teaches a method and system for delivering data from a server object to a client object

using a non-proprietary data transfer protocol. Client objects periodically request to view uniform resource locator that is aliased to a servlet. For each request, the servlet retrieves and delivers a video image to the requesting client object. See *Densmore*, Abstract.

The Office Action alleges that *Densmore* teaches sending a request from a client to the web server, the request identifying requested content and including addresses for a plurality of code modules needed to service the request. The cited portion of *Densmore* states:

> A servlet **200** is a dynamically loadable object that extends the functionality of the server computer **202**. The client computers **204**, **206** access the servlet **200** in related, but different ways. The first client computer **204** accesses the servlet **200** via an applet **207** that is embedded in a web page, which is viewed by a web browser **208**. The applet **207** is a program that is run within the browser **208**. Thus, the applet **207** is similar to the servlet **200** in that both extend the functionality of their host computers. The second client computer **206** also accesses the servlet **200** via a web browser **210**, but does so by pointing the browser **210** to a uniform resource locator that is aliased to the servlet **200**.

Densmore, col. 4, lines 42-53.

> The interactions between the browser **208** and the server software **226** are shown in **FIG. 3**. The process begins when the browser accesses a webpage (i.e., snaplet.html) stored on the server computer **202** (**302**). The browser loads the web page, which contains the applet **207** (**304**). The browser then launches the applet **207**, which requests images by accessing the servlet's URL (**306**, **308**). In turn, the server software **226** invokes the servlet **200** to satisfy the browser's **208** request (**310**). If this is the first request, the server software designates the data source and the frame creation rate, or capture rate, before returning the current image to the browser (**312**, **314**, **316**, **318**, **320**). On subsequent requests, the current image is returned immediately (**312**, **318**, **320**). By making repeated requests to the servlet's URL, numerous images are delivered to the browser (**308**, **318**, **320**).

Densmore, col. 6, lines 36-50. As clearly shown in the cited portions, *Densmore* teaches a single servlet and a plurality of requests for the servlet. *Densmore* does not teach or

suggest "sending a request from a client to the web server, the request identifying requested content and including addresses for a plurality of code modules needed to service the request," as recited in claim 1.

Gosling teaches an apparatus and method for processing servlets in which a specified servlet object corresponding to a request may be uploaded from a remote server to the server receiving the request. The specified servlet object is then executed to obtain dynamically generated information corresponding to the request. See *Gosling*, page 2, lines 29-34. *Gosling* states:

> A server administrator may specify that part of the client request is the name of the servlet, as found in an administered servlets directory. At many sites, that directory would be shared between servers which share the load of processing for the site's clients. Some servers may be able to automatically invoke servlets to filter the output of other servlets, based on their administrative configuration. For example, particular types of servlet output may trigger post-processing by other servlets, perhaps to perform format conversions. Properly authorized clients may specify the servlet to be invoked, without administrative intervention.

Gosling, page 4, lines 10-15. Thus, in *Gosling*, a client request may specify a servlet to provide output for a request. In *Gosling*, the client may also somehow specify a servlet to be invoked to perform format conversions.

At best, *Gosling* teaches that a request may specify a first servlet to provide dynamically created output and *Gosling* also suggests that a client may somehow specify a second servlet that may be invoked to perform conversion on the output of the first servlet. However, *Gosling* does not teach or suggest a client sending a request that identifies requested content and includes addresses for a **plurality** of code modules needed to service the request and the server **applying the plurality of code modules sequentially** to the requested content, as recited in representative claim 1, as amended. Therefore, *Gosling* does not make up for the deficiencies of *Densmore*.

The applied prior art fails to teach or suggest each and every claim limitation; therefore, the proposed combination of *Densmore* and *Gosling* does not render claim 1 obvious. Claims 13, 17, 23, and 28 recite subject matter addressed above with respect to

claim 1 and are allowable for the same reasons. Since claims 2-8, 14-16, 18, 24-27, and 29 depend from claims 1, 13, 17, 23, and 28, the same distinctions between *Gosling* and the invention recited in claims 1, 13, 17, 23, and 28 apply for these claims. Consequently, it is respectfully urged that the rejection of claims 1-8, 13-18, and 23-29 is overcome.

Moreover, the Office Action may not use the claimed invention as an "instruction manual" or "template" to piece together the teachings of the prior art so that the invention is rendered obvious. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). Such reliance is an impermissible use of hindsight with the benefit of applicant's disclosure. *Id.* Therefore, absent some teaching, suggestion, or incentive in the prior art, *Densmore* and *Gosling* cannot be properly combined to form the claimed invention. As a result, absent any teaching, suggestion, or incentive from the prior art to make the proposed combination, the presently claimed invention can be reached only through an impermissible use of hindsight with the benefit of Applicant's disclosure a model for the needed changes.

With respect to claim 9, the Office Action states:

> As per claim 9, Densmore discloses a method for enabling a web client to add functionality to a web server on an as-needed basis, comprising the steps of:
>
> - receiving a request from a client, the request identifying a code module required to process the request (col. 4, lines 42-53, and col. 6, lines 36-50).
>
> However, Densmore does not explicitly discloses:
>
> - responsive to a determination that the code module is not available at the web server, uploading a code module from the client to the web server; and
> - at the web server, using the uploaded code module as needed to service a given request from the web client.
>
> Gosling discloses a method and apparatus for operating a local server computer of a client-server network comprising:
>
> - responsive to a determination that the code module is not available at the web server, uploading a code module from the client to the web server (pg. 3, lines 25-30); and
> - at the web server, using the uploaded code module as needed to service a given request from the web client (pg. 3, lines 25-30).
>
> Therefore, it would have been obvious to one of ordinary skill in the art to combine the teachings of Densmore and Gosling by determining if the code module is unavailable at the web server and uploading the code

module to service the request allowing a user to receive the requested content in a timely and efficient manner.

Office Action, dated September 10, 2003. Applicant respectfully disagrees. The Office Action alleges that *Gosling* teaches uploading a code module from the client to the web server. The cited portion of *Gosling* states:

> Figure 3 is a general illustration demonstrating additional features of the invention. Figure 3 illustrates a local server computer 24A which receives a request from a client computer (not shown) over transmission channel 26. The web server 46 determines that dynamically generated information from a servlet object is required. In this case, the servlet object is not initially on the local server computer 24A, thus it is uploaded by the local server computer 24A from a remote server computer 24B using communication link 26. In the example of Figure 3, servlet 56P is passed from the remote server computer 24B to the local server computer 24A.

Gosling, page 3, lines 25-30. While *Gosling* does indeed teach uploading a servlet from a **remote server** to a local server, *Gosling* does not teach or suggest uploading a servlet from a **client** to the local server, as alleged in the Office Action. The applied references, taken alone or in combination, fail to teach or suggest each and every claim limitation. Therefore, the proposed combination of *Densmore* and *Gosling* does not render claim 9 obvious.

Claim 20 recites subject matter addressed above with respect to claim 9 and is allowable for the same reasons. Since claims 10-12, 21, and 22 depend from claims 9 and 20, the same distinctions between *Densmore* and *Gosling* and the invention recited in claims 9 and 20 apply for these claims. Consequently, it is respectfully urged that the rejection of claims 9-12 and 20-22 is overcome.

Therefore, the rejection of claims 1-18 and 20-29 under 35 U.S.C. § 103 is overcome.

II. Conclusion

It is respectfully urged that the subject application is patentable over the prior art of record and is now in condition for allowance.

The examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: January 8, 2004

Respectfully submitted,

Stephen R. Tkacs
Reg. No. 46,430
Carstens, Yee & Cahoon, LLP
P.O. Box 802334
Dallas, TX 75380
(972) 367-2001
Agent for Applicants